SF 298 MASTER COPY KEEP THIS COPY FOR REPRODUCTION PURPOSES

REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 1998	Technical - 98-06

4. TITLE AND SUBTITLE
A Genetic Algorithm for Variable Knot Spline Fitting via Least Squares

5. FUNDING NUMBERS
DAAH04-96-1-0082

6. AUTHOR(S)
Jennifer L. Pittman

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(ES)
Center for Multivariate Analysis
Dept. of Statistics
417 Thomas Bldg.
Penn State University
University Park, PA 16802

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
ARO 35518.33-MA

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release: distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

In this report we shall describe a method for fitting variable knot spline models to noisy univariate data which uses a genetic algorithm to optimize over the number and location of the knots. For a fixed number of knots, the location of the knots is chosen to minimize the sum of squares error; the appropriate number of knots is determined by the adjusted GCV criterion of Luo and Wahba (1997). The objective is to find the model which minimizes RSS/df, where the degrees of freedom are inflated to reflect the adaptive nature of the knot search (i.e., selection of basis functions). We justify theoretically that our algorithm will converge to the variable knot model which optimizes the model fitting criterion, given that this model is contained in the search space. A modified bootstrap technique is used to obtain pointwise standard errors for models obtained by the GA method. Experimental results comparing the performance of the proposed algorithm to those obtained using the non-linear optimization technique of Schwetlick and Schütze (1995), the genetic algorithm proposed by Manela et. al. (1993), and the method of Luo and Wahba (1997) are presented. We also discuss the extension our technique to related problems.

14. SUBJECT TERMS
Genetic algorithms, variable knot splines. least squares, model selection, statistical data analysis

15. NUMBER IF PAGES
25

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Standard Form 298 (Rev. 2-89)

A GENETIC ALGORITHM FOR VARIABLE KNOT SPLINE FITTING VIA LEAST SQUARES

Jennifer L. Pittman

Technical Report 98-06

May 1998

Center for Multivariate Analysis
417 Thomas Building
Penn State University
University Park, PA 16802

Research work of author was supported by the Army Research Office under Grant DAAHO4-96-1-0082. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

Technical Report No.: MIU/TR/01/98

A Genetic Algorithm for Variable Knot Spline Fitting via Least Squares

Jennifer L. Pittman*

Machine Intelligence Unit
Indian Statistical Institute
203 B.T. Road
Calcutta - 700035, India

March 17, 1998

*The work herein was supported by the U.S. Army Research Office under Assert grant # DAAG55-97-1-0219. The author is a research scholar at the Center for Multivariate Analysis, Department of Statistics, 418 Thomas Building, Pennsylvania State University, University Park, PA - 16802, USA

Abstract

In this report we shall describe a method for fitting variable knot spline models to noisy univariate data which uses a genetic algorithm to optimize over the number and location of the knots. For a fixed number of knots, the location of the knots is chosen to minimize the sum of squares error; the appropriate number of knots is determined by the adjusted GCV criterion of Luo and Wahba (1997). The objective is to find the model which minimizes RSS/df, where the degrees of freedom are inflated to reflect the adaptive nature of the knot search (i.e., selection of basis functions). We justify theoretically that our algorithm will converge to the variable knot model which optimizes the model fitting criterion, given that this model is contained in the search space. A modified bootstrap technique is used to obtain pointwise standard errors for models obtained by the GA method. Experimental results comparing the performance of the proposed algorithm to those obtained using the non-linear optimization technique of Schwetlick and Schütze (1995), the genetic algorithm proposed by Manela et. al. (1993), and the method of Luo and Wahba (1997) are presented. We also discuss the extension our technique to related problems.

Key Words: Genetic algorithms, variable knot splines, least squares, model selection, statistical data analysis.

1 Introduction

Suppose we are asked to analyze a set of N measurements $\{(x_i, y_i) : i = 1, \ldots, N\}$ in $\mathcal{R}^2$, $a \leq x_1 < x_2 < \cdots < x_N \leq b$, where the system which generated the data can be described by an equation of the form $y_i = f(x_i) + \epsilon_i$ for some unknown function f. The ϵ_is are assumed to be independent and follow some distribution with mean zero. One approach to this problem is to attempt to replace the noisy and/or complex relationship which the data represent by something simple but reasonable which captures the nature of the dependence of y on x. To achieve this goal we assume that f can be approximated by a continuous piecewise polynomial which satisfies certain continuity conditions on its lower order derivatives. A natural measure of the quality of such an approximation is a statistic based on the sum of the squares error adjusted to avoid 'overfitting' of the data. In this context, a rich, flexible class of models is the space of spline functions $\mathcal{S}_{m,\mathbf{t}}$ of order m with some knot sequence $\mathbf{t}$.

The choice of knot sequence $\mathbf{t}$, in terms of both the number of knots and their placement, can be crucial for both the shape and the quality of a spline fit [1]. By performing an optimization over the number and location of the knots, as well as the spline coefficients, the model fit can in general be significantly improved. However, determining the best model over this space of variable knot splines is a very poorly behaved non-linear optimization problem [2].

In this paper we propose a method which uses genetic algorithms to determine the number and location of the knots. For a fixed number of knots the location of the knots is chosen to minimize the sum of squares error; the appropriate number of knots is determined by selecting the model which minimizes this error adjusted for the number of knots or 'degrees of freedom' used in the fit. Although we consider only least squares splines, our method can be generalized to spline fitting by penalized least squares or more robust criteria.

Recent contributions to the problem of fitting variable knot splines have been made by several authors. Larson [3] considers linear least squares splines with unknown knot locations while Hu [4] presents an algorithm similar in spirit to that of de Boor [5] which seeks an optimal knot distribution. Schwetlick and Shültze [6] have devised a nonlinear optimization routine which searches for the optimal least squares (LS) knot placement. The number of knots is chosen as the fewest number necessary to model the data within some estimate of the noise level. From the literature on genetic algorithms, Manela et. al. [7] have applied genetic algorithms to penalized least squares spline fitting with optimization over the number of knots, the order of the spline, and the smoothing parameter. They use the method of Dierckx [8] to get a reasonable knot distribution for a fixed number of knots; his recommendations on possible values for the order and number of knots are used to constrain the search space.

A general discussion of methods for approximation by variable knot splines and the motivation for the proposed method are provided in Section 2. Section 3 contains an overview of genetic algorithms as a tool for finding a near optimal solution to the present problem . A specific formulation of this problem followed by a presentation of the GA method and a justification of its convergence are given in Section 4. Implementation details, proposed experiments, and expected results are discussed in Section

5. We conclude with general comments and directions for future research in Section 6.

Only a brief introduction to spline functions will be presented here; an excellent introduction to spline theory and their applications to data analysis can be found in de Boor [5], Wahba [9], Wegman and Wright [10], and Eubank [11]. A nice overview of algorithms for 'free' knot splines similar to the one given below can be found in Schwetlick and Schültze [6].

2 Existing Methodology

For a fixed knot sequence, numerically stable algorithms for the computation of the LS spline model can be found in the literature; see, e.g., de Boor [5]. However, as noted earlier, 'freeing' the knots in the sense of optimizing their placement and number yields a superior spline model in most cases. In essence, the number of knots can be viewed as equivalent to the window size in a smoothing model - more knots leads to more smoothing. 'Freeing' the knots is comparable to using a smoother whose window size is *locally adaptive* - allowing a smaller window size (the placement of more knots) in areas where more flexibility is needed, as in places of peaks or high curvature, and a larger window size (the placement of less knots) in areas of relatively low variability. In this way the number of knots and their placement can make a difference in terms of quality of fit [12]. Hence the selection of a least squares spline model should involve some adaptive procedure for determining the number and location of the knots, i.e., the knot sequence $\mathbf{t}$.

Some of the earliest attempts at fitting variable knot splines include those of Bellman and Roth [13] for linear splines and Hawkins' [14] method based on dynamic programming. These have been followed by algorithms which iteratively add or delete knots but do not attempt optimal knot placement, only a distribution for the knots which is in some sense optimal. Included here is de Boor's algorithm NEWKNOT [5], the method of Agarwal and Studden [15] for linear splines, and the recent contribution of Hu [4]. Manela et. al. [7] avoid the iterative addition and deletion of knots by using genetic algorithms to fit penalized LS splines. They optimize the choice of the smoothing parameter and allow the order of the spline and the number of knots to vary (within certain constraints); for a fixed number of knots they utilize Dierckx's algorithm [8] to determine the knot distribution.

There are also nonlinear optimization routines which attempt to optimize knot placement with respect to LS error for a fixed, given number of knots. de Boor and Rice [16] and Jupp [17] consider LS cubic spline approximation and, beginning with an initial knot sequence, search for the best knot placement by Gauss-Newton methods. The algorithm of Dierckx [18] is similar but employs the Fletcher/Reeves cg method. Schwetlick and Schütze [6] have presented a nice extension of these works based on generalized Gauss-Newton methods which allows for the optimal placement of a subset of the knot sequence. They include a knot removal strategy (see Lyche and Mørken [19]) to search for the model which approximates the data to within its estimated noise level using the fewest number of knots.

Variable knot splines have been suggested in the statistical literature for several data analytic applications, the most predominant of which has been additive modeling. We shall only mention a few selected additive modeling methods with respect to regression spline basis selection; see Hastie and Tibshirani [12] for a more thorough review of additive models. With regression splines and additive modeling, the knot locations are restricted to the set of design points and the problem of finding the optimal set of knots is approached from the perspective of model selection. The proper choice of knot sequence is usually determined by a stepwise addition and/or deletion procedure where only models whose size is within a certain range are considered. The appropriate model is taken as the one which minimizes a criterion based on, for example, generalized cross-validation (GCV) [20] or Mallows' C_p statistic [21]. Alternatively, a generalized backfitting approach similar to that of Breiman [22] could be devised where knot selection is incorporated into the iterative process of determining the current choice of each smoothing matrix. This possibility will not be pursued here.

Methods which begin with many knots and then delete those which are least important with respect to approximation and/or prediction error include those of Smith [23] and Lyche and Mørcken [19]. Friedman and Silverman's [24] algorithm TURBO does stepwise addition of knots to get a series of models; of these, the model which minimizes a GCV criterion is chosen. Knots are then deleted from this model to achieve the final fit. In MARS, Friedman [25] utilizes a stepwise knot addition/deletion strategy for linear splines. His GCV criterion includes a 'cost' term to adjust for the adaptive nature of the knot selection, as suggested by Hastie [26]. Breiman's [27] DK/CV algorithm is based on Smith's work; it uses knot deletion combined with a CV criterion for model selection. His work has shown DK/CV to be superior on small, noisy datasets to ACE [28] which uses a backfitting approach. A recent contribution to additive modeling has been made by Luo and Wahba [29] with their algorithm HAS. They perform forward knot selection via GCV with a 'cost' term but replace backward deletion by ridge regression to reduce model variability and improve numerical stability.

Unlike the algorithms mentioned, the proposed method combines the following:

- The knots are not restricted to the class of design points and for a given number of knots we determine the knot sequence which minimizes the sum of squares error. Hence we simultaneously optimize the choice of model coefficients as well as the number of knots and their placement. For implementation reasons we restrict ourselves to splines with simple knots.

- As opposed to utilizing a stepwise procedure, which is known to be suboptimal, we allow a genetic algorithm to search over the space of models with k interior knots, $k \leq k_{\max}$, and use a model selection criterion based on GCV to choose the model which minimizes error yet contains an appropriate number of knots.

- It has been noted by Jupp [17] that with 'free' knot LS splines, the error function is nonconvex in the knot sequence $\mathbf{t}$. Hence nonlinear optimization algorithms which use generalized Gauss-Newton and/or conjugate gradient methods can converge to local minima or saddlepoints. In contrast, it has been shown theoretically (see Section 4.5) that if the global optimum is contained in the search space then the proposed genetic algorithm will converge to that global optimum regardless of the shape of the function being optimized. It should also be noted

that the result of non-linear optimization often depends on the choice of initial knot sequence whereas genetic algorithms are relatively independent of initial parameter estimates.

Before discussing the details of the proposed method we will give a general overview of genetic algorithms as a tool for optimization.

3 Genetic Algorithms

Genetic algorithms are stochastic search methods which provide a near optimal solution to the evaluation function of an optimization problem [30]. Originally developed by Holland [31], they can be used to search complex, multimodal surfaces via steps which have been designed to mimic the processes of natural genetic systems. They work simultaneously on a number of possible solutions to help prevent the algorithm from getting stuck in a local optimum. With little adjustment - in most cases by only redefining the fitness function - GAs can be applied to a wide range of modeling problems. Thus it is possible to use the same basic algorithm to, for example, fit piecewise polynomials satisfying different optimization criteria to the same dataset. Situations related to various research areas - including scheduling, classification, and pattern recognition - in which the effectiveness of GAs has been demonstrated can be found in the literature [32]. Some good references on GAs include Goldberg [33] and Holland [34].

To find a near optimal solution to a given problem, a genetic algorithm needs only (1) a solution space representation, (2) a choice of operators for performing the search, and (3) a criterion or fitness function to determine the value of a proposed solution. The solution space representation is used to encode each candidate solution as a string or chromosome; a set of such chromosomes is called a *population.* Starting with a randomly generated population, at each iteration the algorithm applies the chosen operators to the given population to yield a new population of strings. The standard operators are selection, crossover, and mutation. Selection gives members of the present population with large fitness values an increased chance of being present in the next population. Crossover or mating allows pairs of strings to combine their better features to create an improved string for the next population. Mutation gives the algorithm an opportunity to branch into previously unexplored regions of the domain space - this helps avoid premature convergence to a local optimum. An elitism step, where the best individual from the present population is included in the subsequent population, is often incorporated. This cycle of operations is repeated until some termination criterion is met, at which time the best string achieved is generally taken as the solution to the optimization problem. The fitness function is used to determine the value of the solution represented by a given string and hence the best solution is represented by the string which maximizes the fitness function.

It has been proven that by including elitism in the genetic algorithm model, one can ensure theoretically that the GA will converge to the global optimum for a sufficiently large number of iterations (Bhandari, Murthy, and Pal [35]). This convergence is independent of the choices for the various algorithm parameters (e.g., crossover and

mutation probabilities, population size). The values of these parameters only influence the rate of convergence.

We refer the reader to Pittman and Murthy [36] for a more detailed overview of the basic GA framework and operators.

3.1 Why GAs?

In the context of the present problem, it is necessary to find not only the proper knot placement but also the proper number of knots. For a fixed number of knots, the sum of squares error is a nonconvex function of the knot sequence. For example, Jupp [17] notes that for the titanium heat dataset of de Boor [5], if one fits 9 variable knot B-splines of order 4 by least squares then the function to be minimized has four stationary points - one interior optimum, two local minima, and one saddle point! Thus a traditional derivative-based approach may get stuck in a local minimum depending upon the choice of the initial knot sequence. With genetic algorithms, as mentioned above, one does have theoretical convergence to the global optimum. In order for this convergence to occur one does not need very good initial estimates of the knot locations.

Determining the proper 'free' knot spline for a given dataset can be viewed as a variable selection problem: given a large set of candidate knot locations and a criterion of fit, find the subset of knots of a certain size which yields the best fit. Exhaustive enumeration is not an option due to the size of the domain space while the various stepwise and stage wise procedures popular in the literature are necessarily suboptimal [30]. Hence genetic algorithms, by performing a directed search over the model space without resorting to stepwise methods, have the potential to better address this sort of problem.

Given the above remarks a note of caution about GAs is in order. Although their convergence to the global optimum has been proven, the choice of the various algorithm parameters - population size, string length, crossover and mutation probabilities - does affect the rate of convergence. The nature of this dependence is generally unknown so the selection of parameter values to achieve the best performance can be a difficult task. GAs are very computer intensive and hence slower than most existing methods. Finally, different runs of the same GA can lead to different results (although the results are usually reasonable solutions). Hence the nature of the current problem may motivate the development of a genetic algorithm based method, but GAs may not be an appropriate optimization tool in other modeling contexts.

4 Formulation of Problem and Proposed Method

We specify the details of the problem of interest and propose a a method based on a modified variable length genetic algorithm for finding a near optimal solution.

4.1 Problem statement

Consider a given univariate dataset $\{(x_i, y_i) : i = 1, \ldots, N\}$, $(x_i, y_i) \in \mathcal{R}^2$ where the $\{x_i\}$ satisfy

$$a \leq x_1 < x_2 < \cdots < x_N \leq b \quad a, b \in \mathcal{R}$$

The observed functional relationship between x and y may be represented as

$$y_i = f(x_i) + \epsilon_i \quad i = 1, \ldots, N$$

where f is a continuous function, $f \in C[a, b]$, and the ϵ_is are independent and identical following a distribution with mean zero. We wish to approximate the function f so that a criterion based on the sum of the squared errors (i.e., sum of squares error) is minimized. We shall represent this criterion as $\phi(\|\mathbf{y} - g(\mathbf{x})\|_{SS})$ for a given approximation g to f, where

$$\|\mathbf{y} - g(\mathbf{x})\|_{SS} = \sum_{i=1}^{N} (y_i - g(x_i))^2$$

4.2 Solution space

Our model space is the class of continuous piecewise polynomials on $[a, b]$ of order m. More specifically, let $\{\tau_i\}_{i=1}^{k+2}$ be a strictly increasing sequence of points in $[a, b]$ with $\tau_1 = a$, $\tau_{k+2} = b$ and k a positive integer, $k \leq k_{\max}$, $k_{\max}$ given. Let $\{P_i\}_{i=1}^{k+1}$ be a sequence of polynomials of order m where, if g is defined as

$$g(x) = P_i(x) \text{ if } \tau_i < x < \tau_{i+1},\ i = 1, \ldots, k+1$$

and $\{v_i\}_{i=2}^{k+1}$ is a given sequence of positive integers, $1 \leq v_i \leq m-1$, then the first v_i derivatives of g at τ_i are continuous. Our model space may be represented as $\mathcal{P}_{m,\tau,v}$ where

$$\mathcal{P}_{m,\tau,v} = \{g : g(x) = P_i(x) \text{ if } \tau_i < x < \tau_{i+1},\ i = 1, \ldots, k+1;\ \text{the first } v_i \text{ derivatives of } g \text{ at } \tau_i \text{ are continuous; } \{\tau_i\}_{i=1}^{k+2},\ \{v_i\}_{i=2}^{k+1}, \text{ and } \{P_i\}_{i=1}^{k+1} \text{ are given sequences, } k \leq k_{\max}\}$$

For computational reasons and to avoid ill-conditioning, each element $g \in \mathcal{P}_{m,\tau,v}$ will be represented as a linear combination of normalized B-spline basis functions of order m. Hence for fixed k and given $\{\tau_i\}_{i=1}^{k+2}$ let $\mathbf{t} = (t_1, t_2, \ldots, t_{n+m})$ be such that

$$t_1 = \cdots = t_m = \tau_1 = a \leq t_{m+1} \leq \cdots \leq b = \tau_{k+2} = t_{n+1} = \cdots = t_{n+m},$$

$n = m + \sum_{i=2}^{k+1}(m - v_i)$, and each τ_i occurs v_i times in $\mathbf{t}$. Then $\mathcal{P}_{m,\tau,v}$ may be expressed as

$$\mathcal{S}_{m,\mathbf{t},v} = \{g \in \mathcal{P}_{m,\tau,v} :\ g = \sum_{j=1}^{n} \alpha_j B_{j,m,\mathbf{t}},\ \alpha_j \in \mathcal{R},\ j = 1, \ldots, n\}$$

where $\{B_{j,m,\mathbf{t}}\}_{j=1}^{n}$ represents the sequence of normalized B-splines of order m with respect to the knot sequence $\mathbf{t}$. The explicit representation of $B_{j,m,\mathbf{t}}$ in terms of m-th

divided differences of the truncated power functions will not be given here; see de Boor [5].

It is known that the problem of best approximation always has a solution in the space of splines with multiple knots but such a solution does not always exist in the space of splines with simple knots [6]. However, for implementation reasons we will restrict ourselves to only simple knot splines, i.e., splines for which $v_i = m-1$, $i = 2, \ldots, k+1$, and $n = m + k$.

If we let $\mathbf{B(t)}$ denote the matrix of dimension $n \times N$ of B-splines of order m with respect to $\mathbf{t}$, so that

$$\mathbf{B(t)} = \{\mathbf{B}_{j,m,\mathbf{t}}(x_i)\}_{\substack{i=1,\ldots,N \\ j=1,\ldots,n}}$$

then the problem of interest can be restated as

$$\min_{\alpha,\mathbf{t}} \phi(\|\mathbf{y} - \boldsymbol{\alpha}\mathbf{B(t)}\|)$$

4.3 The genetic algorithm model

The solution space under consideration contains models of various sizes, i.e., the number of knots is allowed to vary. Hence when we represent each model as a string or chromosome where each parameter is represented by a certain number of characters, the resulting strings will be of different lengths (see below). The genetic algorithm we employ must be capable of handling populations where the string length is not constant. One such algorithm is the variable length genetic algorithm (VLGA).

4.3.1 Variable Length Genetic Algorithm (VLGA)

The variable length genetic algorithm was designed by Bandyopadhyay, Murthy, and Pal [37] for optimizing the number of hyperplanes needed to classify patterns in a multidimensional feature space. It is able to consider solutions with varying numbers of planes by using strings of varying lengths. For example, a set of h hyperplanes, $1 \leq h \leq h_{\max}$, $h_{\max}$ known, could be represented by the string

$$S = (\gamma_1, \gamma_2, \ldots, \gamma_{h*L}); \quad \gamma_i \in \mathcal{A} \ \ \forall\, i = 1, \ldots, h * L$$

where $\mathcal{A}$ is the set of possible character values and L characters are used to represent each plane. For example, if the characters of the strings are bits, then $\mathcal{A} = \{0, 1\}$. Hence the string length is not fixed but *variable* - for a given $h \in \mathcal{H} = \{1, \ldots, h_{\max}\}$, the string length is $h * L$. The algorithm includes the basic stages - selection, crossover, and mutation - as well as elitism. The elitism stage ensures that the algorithm will converge to an optimal solution, theoretically (see Section 4.5.3).

Modified versions of crossover and mutation were designed to handle strings of varying lengths. In the following we assume that the characters of the strings are bits.

- **crossover**

Two strings are chosen at random for single point crossover. If two strings of different lengths are chosen, then the shorter string is padded with #s (a place-holder) so that the lengths of the two strings are equal. If crossover is performed, the hyperplanes in each resultant string are either *complete* (all 0s and 1s or all #s) or *incomplete*. Let h be a hyperplane and define

$$p_{comp} = \frac{\text{number of \#s in representation of h}}{\text{number of bits in representation of h}}$$

For each incomplete hyperplane, with probability p_{comp} all #s are converted to 0s or 1s (for each bit the choice of 0 or 1 is made at random); else all bits are set to #s. Once all of the hyperplanes are complete, those hyperplanes consisting only of #s are shifted to the end of the string.

As an example, suppose $h_{\max} = 3$, $L = 4$, and the pair of strings chosen for crossover is

0 1 0 1 1 1 and 1 0 0 0 1 0 1 0 1

The first string is padded with #s so that both strings have length 9, yielding

0 1 0 1 1 1 # # # and 1 0 0 0 1 0 1 0 1

If the position chosen for crossover is between the seventh and eighth bits, then after crossover the resulting strings are

0 1 0 1 1 1 # 0 1 and 1 0 0 0 1 0 1 # #

Each string contains one incomplete hyperplane; with probability $p_{comp} = 1/3$ the bits in the incomplete hyperplane in the first string are converted to 0s or 1s, and with probability $p_{comp} = 2/3$ the bits in the incomplete hyperplane in the second string are converted to 0s or 1s. If we assume that the bits in the incomplete hyperplane in the first string were all set to #s while the bits in the incomplete hyperplane in the second string were converted to 0s or 1s, then the resulting strings may look like

0 1 0 1 1 1 # # # and 1 0 0 0 1 0 1 0 0

- **mutation**

Mutation can increase or decrease the string length. All strings are initially padded to be of maximum length ($h_{\max} * L$). Conventional mutation is applied to each defined bit (i.e., 0 or 1) with probability p_{m_1}; if a bit is not mutated then it is set to # with probability p_{m_2}. Each undefined bit (#) is mutated to a defined value with probability p_{m_3}. Incomplete hyperplanes which result

are handled as in crossover (see above). If a string of all #s is generated, it is assigned a minimum fitness value and subsequently removed by selection.

The mutation probability may vary over iterations, initially taking a high value (close to 0.5), then decreasing to a prespecified minimum ($\approx 1/M$, where M represents the population size), then increasing again in the later stages of the algorithm (see [38]). When the algorithm has little knowledge of the search space, a high mutation probability encourages it to explore its domain. As the number of iterations increases the algorithm will move towards a solution so the mutation probability is decreased, allowing for a search in the vicinity of that solution. However, this solution may not represent the global optimum, so the mutation probability is again increased to expand the search. In this way premature convergence to a suboptimal solution may be avoided.

The fitness criterion was defined so that the set of hyperplanes with the maximum fitness value would (1) have the minimum number of misclassifications of any set of h hyperplanes, $h \in \mathcal{H}$, and (2) have the fewest number of hyperplanes of all sets of hyperplanes satisfying (1). Hence the fitness function could be represented as

$$fit(S) = (N - miss_S) + \frac{h_S}{h_{\max}}$$

where N is the number of data points, $miss_S$ is the number of points misclassified by the set of hyperplanes represented by S, h_S is the number of hyperplanes represented by S, and $h_{\max}$ is the maximum possible number of hyperplanes. Hence maximizing the fitness function will yield a set of hyperplanes which is parsimonious yet minimizes the number of misclassifications.

Instead of representing sets of hyperplanes, our strings will represent variable knot splines. For a spline with $n+m$ knots, since the first m knots are placed at a and the last m knots are placed at b, it is necessary to encode only the remaining $n-m(=k)$ interior knots. If l_k characters are used to represent each knot location t_i then a set of k interior knots may be represented by the string

$$S = (\gamma_1, \gamma_2, \ldots, \gamma_{k*l_k}); \quad \gamma_i \in \{0,1\} \quad \forall\, i = 1, \ldots, k * l_k$$

Given S the corresponding least squares spline of order m can be computed using existing algorithms (see Section 5). The least squares solution is unique if and only if that matrix of B-spline coefficients has full rank, i.e., if and only if $\|f\|_2 = (\sum_i (f(x_i)^2)^{1/2}$ is a norm on $\mathcal{S}_{m,\mathbf{t}}$. One way to ensure that $\|\cdot\|_2$ is a norm is provided by the *Schoenberg-Whitney Theorem*:

Theorem 4.1 [Schoenberg – Whitney [*39*]] The matrix $\mathbf{B(t)}$ is invertible if and only if there exists a subset $\{x_{i_j}\}$ of $\{x_i\}$ such that

$$B_{j,m,\mathbf{t}}(x_{i_j}) \neq 0, \; j = 1, \ldots, N$$

i.e., if and only if $t_i < x_{i_j} < t_{i+m} \; \forall\, i$.
♠

We enforce the above condition, which is equivalent to setting the minimum span of each basis function to one observation.

The proposed algorithm does include an elitist stage, as was done in the VLGA, to ensure algorithm convergence to an optimal solution. We now discuss the choice of fitness function.

4.4 Fitness function

The VLGA fitness function is based on two criteria - minimization of the number of misclassifications and the number of hyperplanes. Similarly, we are interested satisfying two criteria - minimizing the least squares error of the fit and the required number of knots (i.e., the required degree of fit). However, since our objective is LS modeling as opposed to classification, we shall use a fitness function based on model selection criteria from the statistical literature. A popular class of criteria for determining the appropriate model g are statistics based on generalized cross-validation [20]. These measures of goodness-of-fit are based on the idea, borrowed from parametric linear regression, of minimizing the residual sum of squares adjusted for the amount of fitting being done by the model, or, in other words, for the increased variance associated with increased model complexity. The amount of fit is represented by the number of 'degrees of freedom' used by the model. In spline fitting, the 'degrees of freedom' of a spline g is defined as some function of the smoothing matrix $\mathbf{S}$ where

$$g(\mathbf{y}) = \mathbf{S}\mathbf{y}$$

Some common definitions [40] are:

1. $df = \mathrm{tr}(\mathbf{S}\mathbf{S}^t)$

   This definition is motivated by the equation for linear models

   $$\sum \mathrm{var}(\hat{y}_i) = p\sigma^2$$

   where $p = df$.

2. $df = \mathrm{tr}(2\mathbf{S} - \mathbf{S}^t\mathbf{S})$

   Note that

   $$E(\mathrm{RSS}) = [N - \mathrm{tr}(2\mathbf{S} - \mathbf{S}^t\mathbf{S})]\sigma^2 + f'(\mathbf{I} - \mathbf{S})^t(\mathbf{I} - \mathbf{S})f$$

   If we are simply smoothing noise, so that $f = 0$, then this definition of df is the expected drop in RSS simply due to overfit. In the case of comparing two different fits this definition is particularly convenient since

   $$E(\mathrm{RSS}_1 - \mathrm{RSS}_2) = [\mathrm{tr}(2\mathbf{S}_1 - \mathbf{S}_1^t\mathbf{S}_1) - \mathrm{tr}(2\mathbf{S}_2 - \mathbf{S}_2^t\mathbf{S}_2)]\sigma^2$$

3. $df = \text{tr}(\mathbf{S})$

   This follows the definition of Mallows' C_p where the factor $2p\hat{\sigma}^2$ is added to the RSS to make is unbiased for the predicted MSE. $\text{tr}(\mathbf{S})$ is popular in the spline literature because, under the appropriate Bayesian assumptions, the posterior covariance of $\hat{\mathbf{y}}$ is $\mathbf{S}\sigma^2$.

Definition 3 is a reasonable choice since

1. For symmetric shrinking smoothers with nonnegative eigenvalues, we have

   $$\mathbf{S}\mathbf{S}^t \leq \text{tr}(\mathbf{S}) \leq \text{tr}(2\mathbf{S} - \mathbf{S}^t\mathbf{S})$$

   (actually, with our model class all three values are the same), and

2. $\text{tr}(\mathbf{S}) = \sum_i \lambda_i$, where λ_i is the ith eigenvalue of $\mathbf{S}$, so this definition has the advantage of being easy to compute.

This choice yields the GCV criteria as originally stated by Craven and Wahba [20],

$$GCV(g) = \frac{\frac{1}{N}RSS_g}{\{1 - \text{tr}(\mathbf{S})/N\}^2}$$

where RSS_g is the residual sum of squares from the fit of the model g to the data and $\mathbf{S}$ is the smoothing matrix corresponding to g. With the given class of models the trace of $\mathbf{S}$ is simply the number of basis functions being fit. Hence $\text{tr}(\mathbf{S}) = n$.

A recent advance in additive modeling, where the focus is on fitting regression splines, is to adjust the GCV criterion by inflating the degrees of freedom to account for the adaptive nature of the search for basis functions. This leads to a statistic of the form

$$GCV(g_n) = \frac{RSS_{g_n}}{(1 - (2 + (n-2)d)/N)^2}$$

as used in MARS modeling [25] with $d = 3$. Here we denote g as g_n to explicitly show the number of basis functions of which g is composed. Luo and Wahba [29] have suggested the use of $d = 1.2$ for cubic spline basis functions. Although the procedure is not stepwise, genetic algorithms also employ an adaptive procedure for the selection of basis functions. For this reason our function $\phi\|\mathbf{y} - g(\mathbf{x})\|$ (see Section 4) will be of the above form. However, the 'best' model will *minimize*, not maximize, a criterion of this form. This can be rectified by defining c to be an appropriately chosen, large constant c and maximizing the function $c - GCV(g_n)$ over the given model space. Hence our fitness function may be expressed as

$$FF(g_n) = c - \frac{RSS_{g_n}}{(1 - (2 + (n-2)d_m)/N)^2} \tag{4.4}$$

where g_n is of order m. For basis functions of order $m = 2$ of $m = 4$ (linear or cubic), the values for d_m suggested above will be employed. Otherwise, an appropriate value for d_m can be found by cross-validation or simulation.

4.5 Approximation and convergence

To justify the method outlined above we briefly discuss some approximation properties of spline functions and the convergence of the proposed genetic algorithm to the global optimum.

Recall that the observed functional relationship between x and y is assumed to have the form $y_i = f(x_i) + \epsilon_i$ for some continuous function f where ϵ_i are *i.i.d.* with mean zero. If our goal is to approximate f then our model class $\mathcal{S}_{m,\mathbf{t}}$ should asymptotically contain elements whose distance from f is arbitrarily small. We shall verify this by stating the following theorems which can be found, along with the corresponding proofs, in de Boor [5].

4.5.1 Distance from splines

We first consider the distance of a function h from $\mathcal{S}_{m,\mathbf{t}}$, where h satisfies various conditions. We defer the discussion of least squares approximation to the next section.

Define $\|h\| = \max_{a \le x \le b} |h(x)|$.

Theorem 4.2 Let $g \in \mathcal{S}_{m,\mathbf{t}}$ and $g \in \mathcal{C}[a,b]$. Let $|\mathbf{t}| = \max_i \Delta t_i$ where $\Delta t_i = t_{i+1} - t_i$. Then

$$\operatorname{dist}(h, \mathcal{S}_{m,\mathbf{t}}) = \min\{\|h - g\| : g \in \mathcal{S}_{m,\mathbf{t}}\} \le c_m \cdot \omega(h; |\mathbf{t}|)$$

where $\omega(h; |\mathbf{t}|) = \max\{|h(r) - h(s)| : r, s \in [a,b], |r - s| \le |\mathbf{t}|\}$ and c_m is a constant whose value depends only on m.
♠

In the above theorem, c_m may be taken as $\lfloor (m+1)/2 \rfloor$.

Hence for any fixed order m, we can approximate h to any degree of accuracy if we are willing to use an arbitrary number of knots. In the case that h is smooth, the above bound can be improved considerably.

Theorem 4.3 Let $\mathbf{t} = \{t_i\}_{i=1}^{n+m}$ satisfy

$$t_1 = \cdots = t_m = a < t_{m+1} \le \cdots < b = t_{n+1} = \cdots = t_{n+m}.$$

Then for $j = 0, \ldots, m-1$ there exists $c_{m,j}$ such that for any $h \in \mathcal{C}^{(j)}[a,b]$,

$$\operatorname{dist}(h, \mathcal{S}_{m,\mathbf{t}}) \le c_{m,j} |\mathbf{t}|^j \omega(h^{(j)}; |\mathbf{t}|).$$

If $j = m-1$, then $\omega(h^{(m-1)}; |\mathbf{t}|) \le |\mathbf{t}| \|h^{(m)}\|$ and the above imply

$$\operatorname{dist}(h, \mathcal{S}_{m,\mathbf{t}}) \le c_m |\mathbf{t}|^m \|h^{(m)}\|.$$

♠

Here $c_{m,j}$ may be defined iteratively as $c_{m,j} = c_m^{(j)} = \prod_{i=0}^{j} c_{k-i}$.

In comparison to the bound of Theorem 4.2, the distance of a smooth function from $\mathcal{S}_{m,\mathbf{t}}$ goes to zero at a rate at least as fast as the jth power of the mesh size $|\mathbf{t}|$.

In practice, the number of interior knots is bounded, i.e., $k \leq k_{\max}$, for some positive integer $k_{\max}$. Hence we must examine how well we can approximate f with a fixed number of knots whose placement is allowed to vary.

Let $\mathcal{S}_{m,n}$ denote all splines of order m with some knot sequence $\mathbf{t}_{i=1}^{n+m}$ satisfying $t_1 = \cdots = t_m = a,\ t_{n+1} = \cdots = t_{n+m} = b,\ n$ fixed.

Theorem 4.4 Assume that the function $h \in C[a,b]$ is m times continuously differentiable at all but finitely many points and

$$\int_a^b |D^m h(x)|^{1/m} dx < \infty$$

then

$$\operatorname{dist}(h, \mathcal{S}_{m,n}) \leq c_m n^{-m} \left(\int_a^b |D^m h(x)|^{1/m} dx \right)^m$$

i.e., the order of approximation is n^{-m}.

♠

This bound comes from using Theorem 4.3 and noticing that if $\tau_{i=1}^{k+2}$ is a breakpoint sequence chosen so that

$$\int_{\tau_i}^{\tau_{i+1}} |D^m h(x)|^{1/m} dx = \frac{1}{k+1} \int_a^b |D^m h(x)|^{1/m} dx, \quad i = 1, \ldots, m$$

then

$$\operatorname{dist}(h, \mathcal{P}_{m,\tau} \bigcap C[a,b]) \leq c_m (k+1)^m \left(\int_a^b |D^m g(x)|^{1/m} dx \right)^m.$$

Since $\mathcal{S}_{m,n} \subset \mathcal{S}_{m,\mathbf{t}}$ for $k = n - m \leq k_{max}$, the above order of approximation is a lower bound on the order of approximation attainable by functions from $\mathcal{S}_{m,\mathbf{t}}$.

From the above theorems one can conclude that $\mathcal{S}_{m,\mathbf{t}}$ is a reasonable approximating space for the given problem.

4.5.2 Least squares

Least squares approximation is suitable when the objective is to recover information about a functional relationship from a set of noisy observations. From the above theorems we know that by allowing for many knots, our model class will contain a function whose distance from f is arbitrarily small. Hence, as $N \to \infty$, fitting by least squares will theoretically yield a model whose distance from f is negligable. This is one consideration which motivated our use of least squares in combination with splines for model fitting.

There are some practical considerations, however. First, one can achieve a least squares model simply by allowing for N knots and interpolating the data. Given a noisy dataset this is something to avoid; thus not only should the number of knots be less than the number of data points but the model fitting criterion should penalize models with many terms to avoid 'overfit'. For a fixed number of knots we are finding the LS solution; the penalty provides a way to determine the 'proper' number of knots. Hence such an adjusted sum of squares is a reasonable choice for our fitness function. Another consideration in criterion selection is that, from a statistical standpoint, we desire that the final model be *parsimonious* - for the same amount of error, we prefer a model with few terms. Thus not only should we choose a model fitting criterion which penalizes models with many terms, but we should also select a model class containing those spline functions which maximize the amount of information contained in each knot. Variable knot splines, in combination with an optimization of knot placement, represent such a class.

The amount of improvement in terms of model fit that one can achieve by allowing the knots to vary is, of course, dependent upon the given dataset. On 'nice' datasets with minimal local variability, one would expect less improvement over a spline model with fixed knots or an optimal knot distribution (see Theorem 4.4) as compared to the improvement one would expect on datasets with local properties where the ability of adaptive knots to work as local bandwidth smoothers can be more advantageous. However, given any dataset, optimizing knot placement is attractive when the number of knots in the final model is of importance. Since our objective is a parsimonious model, optimizing knot placement is of key importance.

4.5.3 Convergence to optimum

For fixed knot splines, the model space is linear and finite dimensional on $[a, b]$. The Schoenberg-Whitney condition (see Theorem 4.1) ensures that the finite 'norm'

$$\|g\|_2 = (\sum_i (g(x_i))^2)^{1/2}$$

is a norm and hence a best approximation s^* from this space to f with respect to $\|g\|_2$ exists. However, the space of variable knot splines (fixed dimension) is nonlinear and hence what constitutes a best approximation is difficult to characterize. It also may not be unique. We define the 'best' approximation or fit as a function which maximizes the given fitness function (Eqn. 4.4). For a fixed number of knots this is equivalent to finding the least squares solution from the class of splines with simple 'free' knots.

As stated in Section 3, Bhandari, Murthy, and Pal [35] have proven theoretically that an elitist genetic algorithm (fixed length strings) will converge to an optimal string as the number of iterations goes to infinity. The two characteristics that are necessary and sufficient for algorithm convergence are

- The optimal string from the present population has a fitness value no less than the fitness values of the optimal strings from the previous populations.
- Each string has a positive probability of going to an optimal string within any given iteration.

By preserving these properties, the variable length genetic algorithm has also been shown to converge to an optimal string [37]. This convergence is independent of the choice of values for the algorithm parameters (population size M, crossover probability p_c, mutation probabilities) although these parameter values do influence the rate of convergence. There is no theory to indicate the number of iterations necessary for convergence. Two popular heuristic stopping rules are

- Execute the process for a fixed number of iterations and report the best string found as the solution.
- Execute the process until the fitness value does not show adequate improvement over a fixed number of iterations, and report the best string found as the solution.

We will employ the first stopping rule in our GA based method.

4.6 Variability of results

For a given dataset our method will yield a variable knot spline model g as an estimate of the function f. It is of interest to estimate the variability of our result, i.e., to calculate a statistic which reflects the error of the model given the choice of modeling technique. Note that because of the stochastic nature of the genetic algorithm, different applications of the above method to the same dataset will yield different results. We also have the additional variability stemming from the adaptive determination of the appropriate set of basis functions. Our estimate of model accuracy should reflect both of these sources of error.

For a fixed knot sequence, pointwise standard errors (or global confidence sets) could be calculated theoretically. If $g = \hat{\boldsymbol{\alpha}}\mathbf{B}$ represents the fitted model, where $\mathbf{B}$ is the matrix of B-spline basis functions (the dependence on the chosen knot sequence has been supressed in the notation)), $\hat{\boldsymbol{\alpha}}$ is the vector of basis coefficients, and σ^2 represents the variance of y_i, $i = 1, \ldots, N$, then pointwise standard errors can be derived as follows [12]:

If

$$\Sigma = \mathrm{cov}(\hat{\alpha}) = (\mathbf{B}^t\mathbf{B})^{-1}\sigma^2$$

then

$$\text{cov}(g) = \text{cov}(\hat{\alpha}\mathbf{B}) = \mathbf{B}\text{cov}(\hat{\alpha})\mathbf{B}^t = \mathbf{B}\Sigma\mathbf{B}^t \tag{4.6}$$

σ^2 can be estimated from RSS; the diagonal of $\text{cov}(g)$ contains estimates of the pointwise variances so the diagonal elements of $(\mathbf{B}\Sigma\mathbf{B}^t)^{1/2}$ are estimates of the pointwise standard errors.

One can incorporate the variability due to the adaptive model search into the estimate of σ^2 by adjusting the degrees of freedom, as described in Section 4.4. However, the error due to the stoachastic nature of the GA has yet to be considered. The nascent state of GA theory suggests that the most feasible way to attain a statistic which also captures this source of error is via simulation, e.g., by bootstrapping or other Monte Carlo methods.

The bootstrap was introduced by Efron [41] as a computational method for determining the accuracy of a parameter estimate. It is particularly useful in situations where assessing accuracy is beyond the existing theory of the estimation method or the problem is too complicated for a traditional statistical analysis. The bootstrap is one way to use computational results in lieu of theoretical analysis to effectively provide a formula for the standard error as a function of the sampling distribution of the data. The basic outline of the bootstrap technique is given below; a similar introduction can be found in Efron and Tibshirani [42].

Let $\mathbf{y} = (y_1, \ldots, y_N)$ be observed values of the random variables $x_1, \ldots, x_N \sim i.i.d\ F$ for some probability distribution F. The basic idea behind bootstrap standard errors is to derive $\hat{\sigma}$ from the empirical probability distribution of F. In other words, if $\sigma(F) = [\text{var}_{F,N}(\hat{\theta}(\mathbf{y}))]^{1/2}$, where $\hat{\theta}(\mathbf{y})$ represents our parameter estimate, then $\hat{\sigma} = \sigma(\hat{F})$ where $\hat{F}$ places probability $1/N$ on each observation. $\hat{\sigma}$ is evaluated by a Monte Carlo algorithm:

1. Draw a large number J of bootstrap samples $\{\mathbf{y}_j^b\}_{j=1}^J$ where each is an independent random sample drawn with replacement from the original sample.

2. For each sample $\mathbf{y}_j^b$, $j = 1, \ldots, J$, calculate $\hat{\theta}_j^b = \hat{\theta}(\mathbf{y}_j^b)$.

3. Calculate the sample standard deviation of $\{\hat{\theta}_j^b\}_{j=1}^J$ given by

$$\hat{\sigma}^b = \left(\frac{\sum_j (\hat{\theta}_j^b - \bar{\hat{\theta}}^b)^2}{J-1} \right)^{1/2} \quad \text{where} \quad \bar{\hat{\theta}}^b = \frac{\sum_j \hat{\theta}_j^b}{J}$$

Essentially we are evaluating a standard deviation by Monte Carlo sampling. One can of course replace standard error with some other measure of error, such as bias or prediction error, and apply the same method. In this case only step 3 in the above must be modified.

The number of bootstrap samples necessary for calculating a standard error is an open question. In general, a rough minimum sample size is $50 \leq J \leq 200$.

The above discussion suggests the following method for estimating pointwise standard errors for our model:

Let g be the model fit to the original dataset (i.e., g is our estimate $\hat{\theta}$).

1. Fix the algorithm parameters (e.g., $k_{\max}$, p_c, mutation probabilities, maximum number of iterations) at those values used in determining g. Choose a value for J, $50 \leq J \leq 200$.

2. Draw J bootstrap samples $\{\mathbf{y}_j^b\}_{j=1}^J$ where each is an independent random sample drawn with replacement from the original sample $\mathbf{y} = (y_1, \ldots, y_N)$.

3. On each sample $\mathbf{y}_j^b$, $j = 1, \ldots, J$, apply the GA method to get an estimate $g_j^b = g^b(\mathbf{y}_j^b)$.

4. For each g_j^b, $j = 1, \ldots, J$, calculate the estimate $\hat{\boldsymbol{\sigma}}_j^b$ of the pointwise standard errors. $\hat{\boldsymbol{\sigma}}_j^b$ is defined as the vector whose elements are the square roots of the diagonal elements of $\mathrm{cov}(g_j^b)$ as defined in Eqn. 4.6 where σ^2 is estimated as

$$\hat{\sigma^2} = \frac{RSS}{(1 - (2 + (n-2)d_m)/N)^2}$$

   (see Section 4.4). Here n is the number of basis functions in $\hat{\mathbf{B}}_j$ where $g_j^b = \hat{\boldsymbol{\alpha}}_j \hat{\mathbf{B}}_j$ and d_m is as defined in Section 4.4.

5. Calculate the sample estimate of the pointwise standard errors, $\hat{\boldsymbol{\sigma}}^b$, given by

$$\hat{\boldsymbol{\sigma}}^b = \frac{\sum_j \hat{\boldsymbol{\sigma}}_j^b}{J}$$

The above method, albeit heuristic, uses the bootstrap technique combined with an adjusted pointwise standard error calculation to attempt to capture the various sources of model variability.

Note that with our method we must run a GA to get each g_j^b so the computational expense of such accuracy estimates is considerable. Whether this expense is prohibitive can only be determined through experimentation.

5 Proposed Experiments

5.1 Methods and implementation

5.1.1 Genetic algorithm

We will apply a VLGA to each dataset with $k_{\max}$ chosen appropriately (for some datasets appropriate values for $k_{\max}$ can be estimated from previous analyses where alternative model fitting methods were used). We expect $k_{\max}$ values in the range of 5 to 10. Binary coding will be used although an alternative coding scheme could be used. Each knot location will be represented by l_k bits. If a given knot t_i is represented by the bit sequence $\gamma_1, \gamma_2, \ldots, \gamma_{l_k}$, then the value of t_i is given by the formula

$$t_i = x_{(1)} + \frac{(x_{(N)} - x_{(1)})}{2^{l_k} - 1} \sum_{i=1}^{lk} \gamma_i \cdot 2^{i-1}$$

Hence l_k determines the number of possible values for each knot and the minimum distance between distinct knots. Clearly the choice of l_k depends on the range of x and the distance between consecutive x values, as well as the choice of $k_{\max}$. A reasonable range for l_k is $6 \leq l_k \leq 15$. The crossover probability p_c will be set at 0.8 and the mutation probability p_{m_1} will vary over iterations as described in Section 4.3.1. The other mutation probabilities associated with the VLGA will be set to reflect random selection. M and the maximum number of iterations, $MaxNit$, will be taken as 100 and 10000, respectively. The choice of order will be made from the set $\{3, 4, 5\}$.

In cases where a comparable method (see below) has previously been applied to one of the experimental datasets described below, the choice of certain parameter values will be made to ease the comparison of results.

For a fixed knot sequence, the least squares B-spline coefficients will be calculated using de Boor's algorithm L2MAIN [5].

5.1.2 Methods for comparison

The performance of our method will be compared to the performance of three existing methods for fitting variable knot splines. Each of these methods incorporates a method for selecting an 'optimal' knot sequence.

- **Schwetlick and Schütze (1995)**

  A non-linear optimization algorithm for least squares spline fitting with 'free' simple knots. Instead of enforcing the Schoenberg-Whitney condition, a regularization term of the form

$$\mu \cdot \frac{1}{2} \int_a^b [s^{(r)}(x)]^2, \quad r \text{ fixed}, \quad r \in \{0, \ldots, m-1\}$$

for a spline model of order m is added to the sum of squares error and the combination of the two terms is minimized (hence the models are smoothing splines). The number of knots to include in the model is determined by a stepwise selection procedure. The objective is to model the data within an estimate of its noise level using as few knots as possible. The required parameters (to be selected by the user) include an initial knot sequence, the order of the spline, the smoothing parameter μ, and an estimate of the noise level of the data.

- **Manela et. al. (1993)**

  A genetic algorithm is used to fit B-spline models according to a penalized least squares criterion based on GCV. The smoothing parameter is optimized while the order of the spline and the number of knots are optimized within the constraints given by Dierckx [8]. For a given number of knots, the knot placement is determined by Dierckx's algorithm [8] which attempts an optimal knot distribution. The usual genetic algorithm parameters (e.g, p_c, p_m, $MaxNit$, string length) must be selected.

- **HAS: Luo and Wahba (1997)**

  A cubic regression spline model is fit to the given data using reproducing kernel basis functions. The knot locations are restricted to a chosen subset of the design points. Knots are initially added to the model by forward selection according to the GCV criterion given in Eqn. 4.4. The resulting model is regularized by a ridge regression step in place of backwards deletion of knots, where the ridge regression parameter is determined by GCV. For simulated data, the performance of the final model is evaluated by calculating the median MSE over replications. The set of candidate knot locations and the possible model sizes are determined by the user.

The parameters for these methods will be selected according to previous applications as well as for the purpose of comparison with the proposed method.

5.1.3 Datasets

The performance of the above methods will be compared on both simulated and real datasets in $\mathcal{R}^2$. The simulated datasets are constructed by generating N observations from a given function corrupted with random noise, i.e., $\epsilon_i \sim i.i.d.\ \mathcal{G}(0,\sigma)$, $i = 1,\ldots,N$, for some distribution $\mathcal{G}$, $\sigma > 0$ given. One of the three simulated datasets is borrowed from Luo and Wahba [29],

- $f_1(x) = \sin(2(4x-2)) + 2\exp(-16x^2) \qquad x \in [0,1]$

with $\epsilon_i \sim i.i.d.\ \mathcal{N}(0, 0.4)$, and one is acquired from Schwetlick and Schütze [6],

- $f_2(x) = 10x/(1+100x^2) \qquad x \in [-2,2]$

with $\epsilon_i \sim$ *i.i.d.* $Unif(-0.05, 0.05)$. For both of these datasets the x_is are equally spaced. The remaining simulated dataset will be generated from

- $f_3(z) = \cos(3\pi z)/(0.5 + 4z^2)$

where $z_i \sim$ *i.i.d.* $Unif(0,1)$ and $\epsilon_i \sim$ *i.i.d.* $\mathcal{N}(0, 0.2)$.

The real dataset is the Imports data set from Hand et. al. [43].

We shall run several replications of each method on the third simulated dataset (with a fixed choice of $\{z_i\}_{i=1}^N$) and compare the median MSE performance of the above methods as was done in [29].

We summarize the above in the following table:

Set #	f	N	ϵ_i
1	$f_1(x) = \sin(2(4x-2)) + 2\exp(-16x^2)$	256	$\mathcal{N}(0, 0.4)$
2	$f_2(x) = 10x/(1+100x^2)$	90	$Unif(-0.05, 0.05)$
3	$f_3(z) = \cos(3\pi z)/(0.5+4z^2)$	100	$\mathcal{N}(0, 0.2)$
4	Imports	30	

Table 5.1.3: Experimental Datasets

For the function f_1 we shall generate 100 bootstrap samples and calculate pointwise standard errors for the fitted GA model via the method described in Section 4.6. These will be used to plot ± 2 standard-error bands with our estimate.

5.2 Expected results

We expect our method to yield comparatively simple yet accurate models for several reasons. First, methods where the knot distribution is optimized are not placing the knots with the objective of minimizing the sum of squares error. By doing so, the GA method should yield more accurate models with respect to this error. Second, stepwise knot selection methods are known to be suboptimal - in the above method this type of search has been replaced by a search which theoretically yields the optimal knot sequence. Finally, the GA method is quite flexibile in terms of model selection since it does not restrict the possible knot locations to a subset of the observations. We would also expect to obtain parsimonious models since by optimizing knot placement we are maximizing the amount of information which can be represented by each knot.

It should be noted that, unlike Manela et. al. [7], we are not optimizing the order of the spline. The prevailing opinion from the spline literature is that knot placement is more critical than the choice of order [1]; for this reason we expect the benefits of optimizing knot placement to outweigh the disadvantage of possibly having a suboptimal choice of order.

These reasons aside, the amount of improvement of the GA model will depend on the properties of the given dataset. As mentioned in Section 4.5, the optimizing of knot placement will be most beneficial when the dataset shows substantial local properties, e.g., high local variability, peaks, and regions of high curvature. If the observed data does not show these types of characteristics, then the additional computational resources required to optimize the knot placement may be viewed as an unnecessary expense. It is also possible that the applicability of the method will be restricted because of high computational cost. In this case the decision to implement the GA method on a given dataset should only be taken after a careful consideration of the required resources and the potential modeling benefits.

6 Conclusions and future research

A method has been proposed for fitting variable knot splines where the number of knots as well as their placement are optimized with respect to an adjusted sum of squares fitting criterion. The optimization of knot placement should lead to improved models, especially for datasets with local properties such a peaks and areas of high variability. The resulting models should also be parsimonious in the sense that each knot may be placed to maximize the amount of information it contains regarding the relationship between the observed variables. The expense of these models in terms of computational resources may, however, outweigh their improved performance. This can only be determined by experiments such as those outlined above.

If the genetic algorithm method does prove to be substantially beneficial, two directions for future research are under consideration: extension to high dimensions and other model selection criteria.

6.0.1 Higher dimensions

Currently the most feasible approach for extension to higher dimensional data appears to be through additive modeling, as was done by Rogers [44] with MARS models [25]. We would like to examine whether replacing stepwise techniques with genetic algorithm optimization for knot sequence selection can lead to improved fitting of other types of additive models; more specifically, HAS models [29] and the polynomial tensor product spline models of Stone et. al. [45].

6.0.2 Model selection criteria

The statistical literature contains a rich class of model selection criteria based on different measurements of model error and degree of model fit. These include statistics based on Mallows' C_p [21], AIC [45], and various GCV criteria ([20], [25], [29]). Our choice of model selection criteria, although supported by statistical arguments, is mainly heuristic. Other data analysts may prefer to use different criteria; for this reason we intend to adapt our procedure for use with other selection criteria.

We also recognize that least squares methods have problems with outliers, so one may prefer to use a more robust criterion. Here the RSS can be replaced by a function of the form

$$\sum_{i=1}^{N} \psi(\frac{y_i - g(x_i)}{\varsigma})$$

where ς is a scale measure, e.g., $\hat{\sigma}$. For a fixed set of basis functions $\{B_j\}_{j=1,\ldots,n}$ this leads to the modified normal equations

$$\sum_{i=1}^{N} B_j(x_i)\psi(\frac{y_i - g(x_i)}{\varsigma}) = 0 \quad j = 1, \ldots, n$$

Popular choices for ψ are the M-estimate [47]

$$\psi(x) = \begin{cases} x & \text{if } |x| \leq c \\ c \text{ sign}(x) & \text{otherwise} \end{cases} \tag{6.0.2}$$

for some constant c or Andrews' [46] sine function

$$\psi(x) = \begin{cases} a^2(1 - \cos(x/a)) & \text{if } |x| \leq \pi a \\ 2a^2 & \text{otherwise} \end{cases}$$

where a is a given constant. Lenth [47] has fit robust cubic splines with ψ as given in Eqn. 6.0.2; the implementation of a similar fitting procedure combined with GA knot selection is a current focus of research interest.

7 Acknowledgements

I would like to acknowledge the many helpful discussions with Professor C.A. Murthy which greatly facilitated the enclosed work.

References

[1] R.L. Eubank, in discussion to J. Ramsay, "Monotone regression splines in action", *Stat. Sci.*, **3**, pp. 425-462, 1988.

[2] G. Wahba, in discussion to J. Ramsay, "Monotone regression splines in action", *Stat. Sci.*, **3**, pp. 425-462, 1988.

[3] H.J. Larson, "Least squares estimation of linear splines with unknown knot locations", *Comp. Statist. and Data Anal.*, **13**, pp.1-8, Jan. 1992.

[4] Y. Hu, "An algorithm for data reduction using splines with free knots", *IMA J. Numer. Anal.*, **13**, pp. 328-343, 1993.

[5] C. De Boor, *A practical guide to splines*, 1st Edition. New York: Springer-Verlag, 1978.

[6] H. Schwetlick and T. Schütze, "Least squares approximation by splines with free knots", *BIT*, **35**, pp. 361-384, Sept. 1995.

[7] M. Manela, N. Thornhill, and J.A. Campbell, "Fitting spline functions to noisy data using a GA", in S. Forrest (ed.), *Proc. of 5th International Conference on Genetic Algorithms*, San Mateo, CA: Morgan Kaufmann, 1993.

[8] P. Dierckx, "An algorithm for smoothing, diff., and int. of experimental data using spline functions", *J. of Comp. and App. Math.*, **1**, pp. 165-184, 1975.

[9] G. Wahba, "Smoothing noisy data with spline functions", *Num. math.*, **24**, pp. 383-393, 1975.

[10] E. Wegman and I. Wright, "Splines in statistics", *JASA*, **78**, pp. 351-365, 1983.

[11] R.L. Eubank, *Spline Smoothing and Nonparametric Regression*, 1st Edition. New York: Marcel Dekker, 1988.

[12] T.J. Hastie and R.J. Tibshirani, *Generalized Additive Models*. London: Chapman and Hall, 1990.

[13] R. Bellman and R. Roth, "Curve fitting by segmented straight lines", *JASA*, **64**, pp. 1079-1094, 1969.

[14] D.M. Hawkins, "On the choice of segments in piecewise approximation", *J. of Inst. of Math. and Apps.*, **9**, pp. 250-256, 1972.

[15] G.G. Agarwal and W.J. Studden, "Asymptotic design and estimation using linear splines", *Commun. Stat. B*, **7**, pp. 309-320, 1978.

[16] C. de Boor and J. R. Rice, "LS cubic spline approximation II: variable knots", Technical Report CSD TR 21, Computer Science Department, Purdue University, 1968.

[17] D.L.B. Jupp, "Approximation to data by splines with free knots", *SIAM J. Numer. Anal.*, **15**, pp. 328-343, 1978.

[18] P. Dierckx, *Het aanpassen van krommen en oppervlakken aan meetpunten met behulp van spline funkties*, PhD thesis, Katholieke Universiteit Leuven, 1979.

[19] T. Lyche and K. Mørken, "A data reduction strategy for splines with applications to the approximation of functions and data", *IMA J. Numer. Anal*, **8**, pp. 185-208, 1988.

[20] P. Craven and G. Wahba, "Smoothing noisy data with spline functions", *Numer. math.*, **31**, pp. 377-403, 1979.

[21] C.L. Mallows, "Some comments on C_p", *Techno.*, **15**, pp. 661-675, 1973.

[22] C. Breiman, "The Π method for estimating multivariate functions from noisy data (with discussion)", *Techno.*, **33**, pp. 125-160, 1991.

[23] P.L. Smith, *Curve fitting and modeling with splines using statistical variable selection techniques*, NASA Report 166034, Langley Research Center, Hampton, VA., 1982.

[24] J. H. Friedman and B. W. Silverman, "Flexible parsimonious smoothing and additive modeling (with discussion)", *Techno.*, **31**, pp. 3-39, 1989.

[25] J. H. Friedman, "Multivariate adaptive regression splines (with discussion)", *Ann. Statist.*, **19**, pp. 1-141, 1991.

[26] T.J. Hastie, in discussion to Friedman and Silverman, "Flexible parsimonious smoothing and additive modeling (with discussion)", *Techno.*, **31**, pp. 3-39, 1989.

[27] C. Breiman, "Fitting additive models to data", *Comp. Statist. and Data Anal.*, **15**, pp. 13-46, 1993.

[28] C. Breiman and J.H. Friedman, "Estimating optimal transformations for multiple regression and correlation (w/ discussion)", *JASA*, **80**, pp. 580-619, 1985.

[29] Z. Luo and G. Wahba, "Hybrid adaptive splines", *JASA*, **92**, pp. 107-115, 1997.

[30] S. Chatterjee, M. Laudato, and L.A. Lynch, "Genetic algorithms and their statistical applications: an introduction", *Comp. Statist. and Data Anal.*, **22**, 6, pp. 633-651, Oct. 1996.

[31] J.M. Holland, *Adaptation in natural and artificial systems*, Ann Arbor, MI: The University of Michigan Press, 1975.

[32] L. Davis (ed.), *Handbook of Genetic Algorithms*, 1st Edition. New York: Van Nostrand Reinhold, 1991.

[33] D.E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Reading, MA: Addison-Wesley, 1989.

[34] J.M. Holland, *Adaptation in natural and artificial systems*, Cambridge, MA: The MIT Press, 1992.

[35] D. Bhandari, C.A. Murthy, and S.K. Pal, "Genetic algorithm with elitist model and its convergence", *Int. J. Patl. Recog. Art. Intell.*, **10**, 6, pp. 731-747, Sept. 1996.

[36] J. Pittman and C.A. Murthy, "Fitting optimal piecewise linear functions using genetic algorithms", *IEEE PAMI*, communicated.

[37] S. Bandyopadhyay, C.A. Murthy, and S.K. Pal, "Pattern classification using genetic algorithms: Determination of H", *Pattern Recognition Letters*, communicated, 1998.

[38] S. Bandyopadhyay, C.A. Murthy, and S.K. Pal, "Pattern classification with genetic algorithms", *Pattern Recognition Letters*, **16**, pp.801-808, August 1995.

[39] I.J. Schoenberg and A. Whitney, "On Pólya frequency functions III: The positivity of translation determinants with an application to the interpolation problem by spline curves", *Trans. Amer. Math. Soc.*, **74**, pp. 246-259, 1953.

[40] A. Buja, T.J. Hastie, and R.J. Tibshirani, "Linear smoothers and additive models (w/ discussion), *Ann. Statist.*, **17**, pp. 453-56, 1989.

[41] B. Efron, "Bootstrap methods: Another look at the Jackknife", *Ann. Statist.*, **7**, pp. 1-26, 1979.

[42] B. Efron and R.J. Tibshirani, "Bootstrap for standard errors, confidence intervals, and other measures of statistical accuracy", *Stat. Sci.*, **1**, pp. 54-77, 1986.

[43] D.L. Hand et. al., *Handbook of small datasets*, London: Chapman and Hall, 1994.

[44] D. Rogers, "G/SPLINES: A hybrid of Friedman's Multivariate Adaptive Regression Splines (MARS) Algorithm with Holland's Genetic Algorithm". In R.K. Belew & L.B. Hooker (eds.), *Proceedings of the Fourth International Conference on Genetic ALgorithms*, San Mateo, CA: Morgan Kaufmann, 1992.

[45] C. Stone, M. Hansen, C. Kooperberg, and Y.K. Truong, "Polynomial splines and their tensor products in extended linear modeling", Tech. Report 437, Dept. of Statistics, Univ. of Cal. at Berkeley, 1995.

[46] D.E. Andrews, "A robust method for multiple linear regression", *Techno.*, **16**, pp. 523-531, 1974.

[47] R.V. Lenth, "Robust splines", *Commun. Stat. A*, **6**, pp. 847-854, 1977.